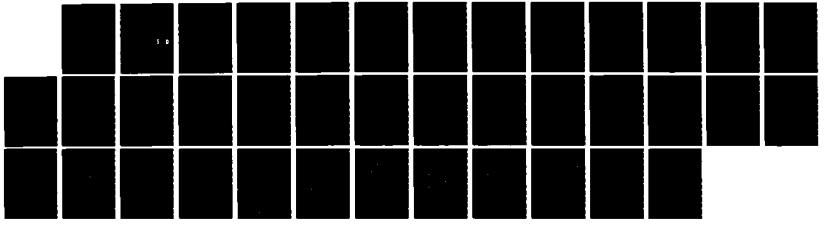
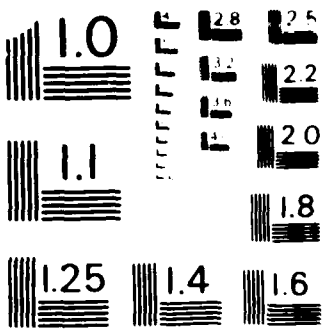
AD-A176 440 THE HOPFIELD MODEL AND BEYOND(U) BROWN UNIV PROVIDENCE RI CENTER FOR NEURAL SCIENCE C M MACHANNAN 15 DEC 86 TR-4 ARO-22000.5-LS DAAG29-84-K-0202 1/1

UNCLASSIFIED F/G 6/4 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8

(2)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

| REPORT DOCUMENTATION PAGE | | READ INSTRUCTIONS BEFORE COMPLETING FORM |
|---|---|---|
| REPORT NUMBER: ARO 22000.5-LS | 2. GOVT ACCESSION NO.: N/A | 3. RECIPIENT'S CATALOG NUMBER: N/A |
| TITLE (and Subtitle): THE HOPFIELD MODEL AND BEYOND | | 5. TYPE OF REPORT & PERIOD COVERED: TECHNICAL |
| | | 6. PERFORMING ORG. REPORT NUMBER: 22000-LS #4 |
| AUTHOR(s): Charles McKay Bachmann | | 8. CONTRACT OR GRANT NUMBER(s): DAAG29-84-K-0202 |
| PERFORMING ORGANIZATION NAME AND ADDRESS: CENTER FOR NEURAL SCIENCE, BROWN UNIVERSITY, PROVIDENCE, RHODE ISLAND 02912 | | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: N/A |
| CONTROLLING OFFICE NAME AND ADDRESS: U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709 | | 12. REPORT DATE: December 15, 1986 |
| | | 13. NUMBER OF PAGES: 35 pgs. |
| 14. MONITORING AGENCY NAME & ADDRESS (If different from Controlling Office) | | 15. SECURITY CLASS. (of this report): Unclassified |
| | | 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE |

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.



17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA

18. SUPPLEMENTARY NOTES

The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Hopfield Model
Widrow-Hoff Algorithm
Stable State
Storage Capacity
Error-Correction
"Unlearning"

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The standard Hopfield Model, (both original and analog) and algorithms to improve its performance are reviewed. An analysis of the model and the modification algorithms is given. Future directions for continuous models which have both large capacity and good error-correcting capabilities are examined.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

# THE HOPFIELD MODEL AND BEYOND

BY

Charles McKay Bachmann

December 15, 1986

Department of Physics
and
Center for Neural Science

Brown University
Providence, Rhode Island 02912

*This work was supported by ARO, Army Research Office Contract #DAAG29-84-K-0202 and ONR, Office of Naval Research Contract #N00014-81-K-0041.

# The Hopfield Model and Beyond


## Abstract

The standard Hopfield model (both digital and analog) and algorithms to improve its performance are reviewed. An analysis of the model and the modification algorithms is given. Future directions for continuous models which have both large capacity and good error-correcting capabilities are examined.

# Chapter 1

# The Discrete Hopfield Model

## 1.1 Introduction to the Model

In 1982, Hopfield [1] proposed a neural model of memory storage and retrieval based on the theory of spin glasses in solid state physics . In the model, neurons are binary-valued threshold units, taking either the value 0 or 1 in one version, or 1 or -1 in an alternative version. This digital restriction of the neurons represents the neuron in two possible states–a 1 represents a neuron that is firing, while a 0 or a -1 , a neuron that is inactive. Mathematically, this corresponds to replacing the experimentally observed neuronal input-output relationship, a graded response which can be characterized by a sigmoid function, with a step-function. The neurons form a single layer and are completely interconnected, with the strength of these connections, or "synapses", given by a correlation matrix formed from the memory states to be stored in the system.

$$w_{ij} = \sum_{s=1}^{m} \mu_i^s \mu_j^s \tag{1.1}$$

Once the layer of neurons is given an input, that is, when the neurons are set to some initial configuration of values, the neurons are updated asynchronously and in a random order. The updating procedure, which is

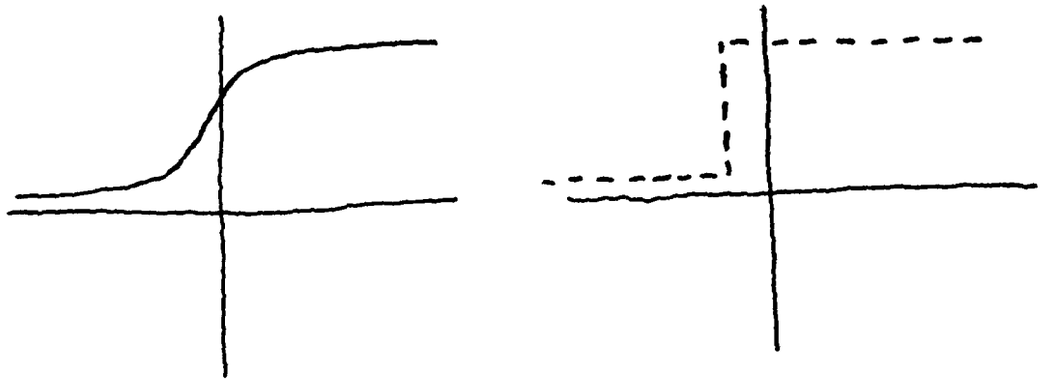


Figure 1.1: Left: Neuronal input-output function: a sigmoid; Right: Hopfield's approximation of the neuronal input-output function: a step-function

given by:[1]

$$\mu_i \rightarrow (\sum_{j=1}^{N} w_{ij}\mu_j) \tag{1.2}$$

amounts to a relaxation process. [2] This follows from the fact that the updating procedure minimizes an associated energy functional, or Liapunov Function. The energy function which is minimized during the relaxation process is:

$$\xi_{\vec{\mu}} = -\frac{1}{2}\sum_{i,j} w_{ij}\mu_i\mu_j \tag{1.3}$$

That $\xi$ is a Liapunov function may be demonstrated as follows.[1] Note that :

$$\Delta\xi = -\Delta\mu_i \sum_{j=1}^{N} w_{ij}\mu_j \tag{1.4}$$

Also observe that :

$$\Delta\mu_i > 0 \Rightarrow \sum_j w_{ij}\mu_j > 0 \Rightarrow \Delta\xi < 0 \tag{1.5}$$

$$\Delta\mu_i < 0 \Rightarrow \sum_j w_{ij}\mu_j < 0 \Rightarrow \Delta\xi < 0$$

$$\Delta\mu_i = 0 \Rightarrow \Delta\xi = 0$$

Thus:

$$\Delta\xi \leq 0, \; and \; \Delta\xi = 0 \; iff \; \Delta\mu_i = 0 \tag{1.6}$$

When this observation is combined with the fact that $\xi$ is a bounded function, the proof that the relaxtion process will descend to a local minimum is complete.

The aim of the Hopfield model is the categorization of an input state according to the stored state to which it is most similar. Given that, if the Hopfield model functioned ideally, the stored states should be the only minima of the relaxation process and should divide the space so that their

[1] Henceforth, we will simply write the function $\theta(arg.)$ to represent $2\theta(arg.) - 1$ when we are dealing with neurons whose values are (+/-)1, and it will be understood to mean the standard Heaviside function when the neurons have the value 0 or 1.

[2] The original model had spins 0 and 1 and no self-connections, but experimental evidence suggests that a system with spins (+/-)1 works better [1] and that adding the self-connections improves the performance still more.[7]

radii of attraction draw the input state to the stored state which it most resembles.

In its original form, the standard Hopfield model, as we have described it above, functions very poorly as a categorizer when $(m/N) \tilde{>} 0.1$.[2] In his 1982 paper, for instance, Hopfield [1] found that when $(m/N) stackrel{\sim}{<}$ .05 to 0.1, the stored states can all be perfectly recalled when presented as input states, that is, they are fixed points, or minima, of the relaxation process. The radius of attraction is also reasonable in this range of (m/N). To be concrete, Hopfield found for (m/N) = 0.05 with N = 30, 90 percent of the random starting states within a radius of 5 hamming units of the stored states relaxed to the target stored state. When (m/N) is above the range 0.05 to 0.1, the attractive capability of the stored states decays rapidly, and the percentage of stored states which are minima also quickly declines. For instance, as a benchmark, at (m/N) = 0.15, Hopfield found only half of the stored states were fixed points.

## 1.2 Methods to Improve the Performance of Hopfield's Model

### 1.2.1 "Unlearning"

Given the limitations of the original model, improvements have been sought. An early approach first tried by Hopfield [2] has been given by him the name of "unlearning", after a term first coined by Crick to explain the biological purpose of sleep in humans and animals as a period during which unneeded information is erased and stored information compacted. In Hopfield's algorithm, a random state is relaxed to a stable state (often a spurious attractor), a correlation matrix is formed from this state, and then an ammount proportional to this is subtracted from the original matrix:

$$w_{ij} \rightarrow w_{ij} - \alpha \mu_i^{relaxed} \mu_j^{relaxed} \tag{1.7}$$

The operation is repeated k times. In simulations of Hopfield's "unlearning", Terry Potter [7] has found that k is optimal for:

$$\alpha k \simeq \frac{1}{2} \tag{1.8}$$

[2] m = the number of stored states. N = the number of neurons.

With "unlearning" the number of stored states that can be correctly recalled approaches the dimensionality, N, and error correction is improved but falls to zero as $m \rightarrow N$.[7]

### 1.2.2 An Alternative Approach to Improvement of the Hopfield Model

Recently, an interesting variation of Hopfield's "unlearning" has been studied experimentally by Terry Potter.[7] The algorithm is a hybrid combining elements of Hopfield's "unlearning" with a modification reminiscent of the Widrow-Hoff algorithm from Linear Filter Theory. As a quick review, we present the rudiments of the Widrow-Hoff algorithm and then examine which elements of it have been carried over into Potter's algorithm.

The goal of the traditional Widrow-Hoff algorithm is to associate pairs of real-valued input, $\vec{x}^k$, with real-valued output, $\vec{y}^k$. A *linear input-output function*, represented by a matrix A, is postulated. The algorithm converges to an optimal matrix $A^*$ by minimizing the mean-square error between the target state and the actual output of the matrix A. For a proof of this, see T. Kohonen(1974).[6] In practice, the derivative of the partial error due to the kth pattern:

$$\frac{\partial E^k}{\partial A_{ij}} = -2(y_i^k - (A\vec{x}^k)_i)x_j^k \tag{1.9}$$

is used as an approximation to the derivative of the total error in a gradient descent algorithm, which has the form:

$$A_{ij} \rightarrow A_{ij} - \alpha \frac{\partial E^k}{\partial A_{ij}} \tag{1.10}$$

The patterns are presented successively, and a modification is made at each step. The motivation for such an algorithm can be depicted graphically: The graph displays the error $E^k$ versus the matrix element $A_{ij}$; the algorithm chooses at each step a better approximation to the global minimum by generating a matrix A which is a better approximation to the minimum of $E^k$. In the limit as the number of presentations increases, the algorithm approaches the global minimum asymptotically, provided that $\alpha$ is chosen sufficiently small [6].

Having described the original Widrow-Hoff algorithm, let us now examine how this has been woven, along with Hopfield's "unlearning", into

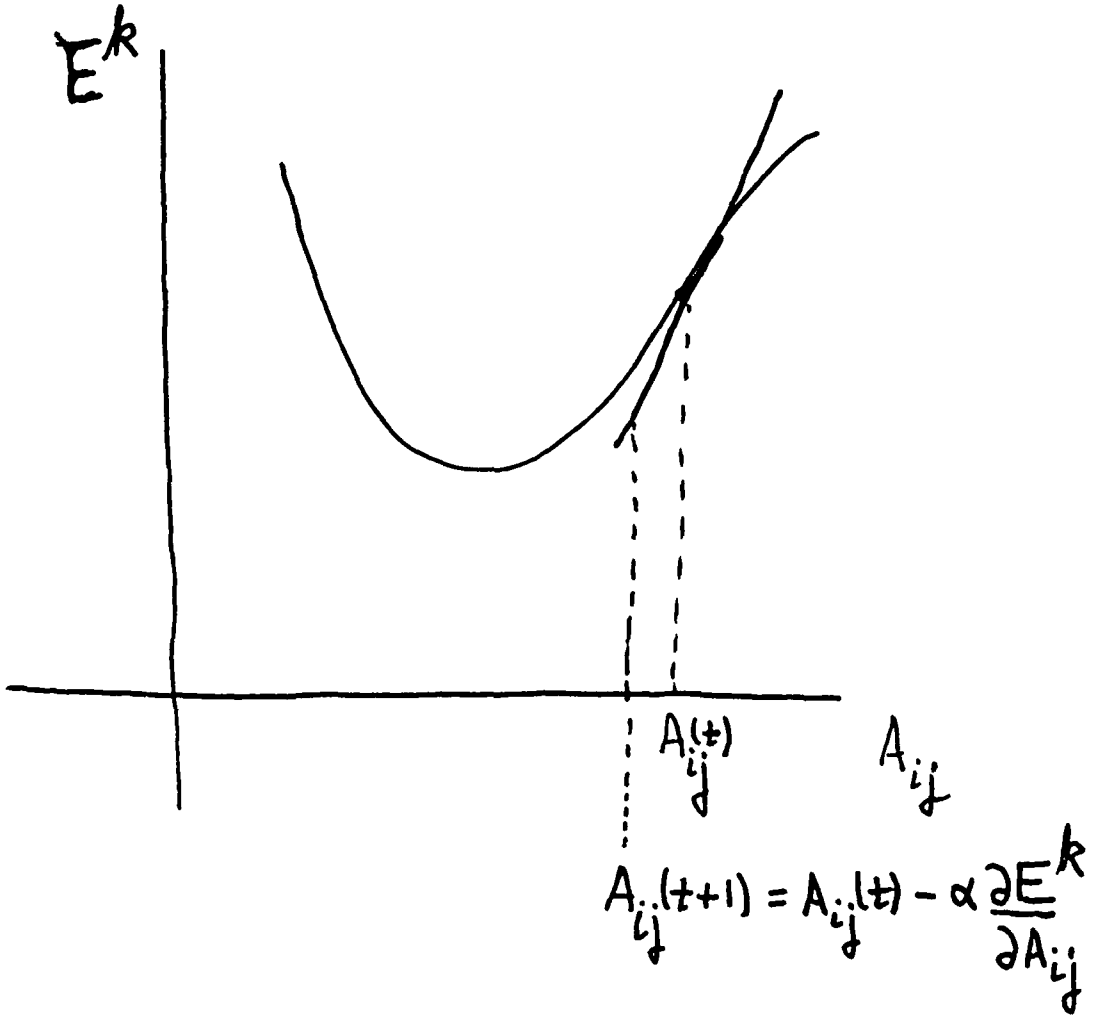



Figure 1.2: Widrow-Hoff minimization

a hybrid algorithm by Potter. There are actually two versions of the algorithm. In one version, following a suggestion by Professor Cooper, the stored states are used as the input states in the relaxation process; for a given stored state, modification is done only if that state is unstable. In a second version of the algorithm, all of the states at a radius of one hamming unit from each stored state are relaxed; a modification is made if the target state misses its target stored state.

The similarity of Potter's algorithm to that of Widrow and Hoff lies in the actual form of the modification used once a state, elected as described above, is relaxed:

$$w_{ij} \rightarrow w_{ij} - \alpha(\mu_i^{target} - \mu_i^{relaxed})\mu_j^{input}(\mu_j^{input} + 1) \qquad (1.11)$$

By analogy with the original Widrow-Hoff equation, 1.10, the result of the entire relaxation process $\mu_i^{relaxed}$ now replaces the output of Widrow-Hoff's linear input-output function, $(A\vec{x}^k)_i$. One should note, however, that the form is not exactly the same as the original Widrow-Hoff because of the presence of the factor $\mu_j+1$. This additional factor and the symmetry of the modification in the indices i and j establishes an additional restriction on the modification criteria mentioned above: no modification will be made to $w_{ij}$(and $w_{ji}$) unless either one or both of the ith and jth elements of the input state has the value one. Potter has found that this additional proviso is an important factor in the effectiveness of his algorithm. There is one more essential difference and that is that the symmetry of the synaptic matrix is preserved by making the same modification to $w_{ji}$ each time a modification is performed on the element $w_{ij}$. This ensures that the relaxation algorithm will continue to descend monotonically: in order to avoid the possibility of limit cycles, the symmetry of the matrix must be preserved. [1]

As an overview of the algorithm, then, the relaxation process as a whole has been embedded in place of the output of the linear operator in the original Widrow-Hoff theory, the overall morphology of the modification has been modified slightly, and additionally, symmetric modifications have been added.

In simulations carried out by Potter [7] using the stored states as the input for the modification, he was able to achieve $\sim N^2$ stable stored states (fixed points). [3] At this density of stored states, the radius of attraction,

[3] a greater number of states was not attempted

however, can not be completely gauranteed for even a radius of one hamming unit: at a radius even so small as this, the percentage of states which converge to the target stored state is $\sim$ 40 to 60 percent. [3]

For simulations in which Potter used all of the input states at a radius of one unit from each of the stored states, he found that for m just below the dimensionality, N , a radius of attraction of one unit hamming unit could be completely gauranteed. Above the dimensionality, the radius of attraction and the percentage of stable stored states decays.

In summary, then, with Potter's algorithm, the capacity of the Hopfield model to generate stable stored states can be vastly improved, but with no radius of attraction. Conversely, with the alternative version of his algorithm, confining the capacity of stored states to be just below the dimensionality, only a severely limited radius of attraction can be constructed around the stored states.

## 1.3 Some Analysis of the Hopfield Model and Potter's Algorithm

To begin with, we note that in the standard Hopfield model, N stable stored states could have been achieved if the coding of the stored states had been selected so that all of the stored states were mutually orthogonal. The proof is as follows. Suppose we choose m mutually orthogonal stored states. Label them by the index s:$1 \cdots m \leq N$. Now, examine what happens when the relaxation process is applied to one of the stored states s':

$$\begin{aligned} \mu_i^{s'} &\rightarrow \theta(\sum_{j=1}^{N} w_{ij}\mu_j^{s'}) \\ &= \theta(\sum_{s=1}^{m} \mu_i^s(\vec{\mu}^s \cdot \vec{\mu}^{s'})) \\ &= \theta(N\mu_i^{s'}) \\ &= \mu_i^{s'} \end{aligned} \tag{1.12}$$

Thus, no matter which neuron is sampled by the updating procedure, it returns the same value for the neuron. Thus, all of the stored states are

[3] provided the self-connections are used; if the self-connections are removed when the synapses are formed, the performance at this radius is even worse.[7]

stable.

Professor Cooper has pointed out that even in a linear system, we should expect this result. The argument is as follows. Suppose, in such a system we choose:

$$w_{ij} = \sum_{s=1}^{m} \frac{\mu_i^s \mu_j^s}{\vec{\mu}^s \cdot \vec{\mu}^s} \tag{1.13}$$

where the vectors $\mu^s$, $s : 1 \cdots m \leq N$, are mutually orthogonal. Such a set can always be generated from a linearly independent set of states via a Gram-Schmidt process. We can rewrite the matrix $w_{ij}$ in terms of an orthonormal set of vectors $\vec{x}^s$ as:

$$w_{ij} = \sum_{s=1}^{m} x_i^s x_j^s \tag{1.14}$$

If the state $\vec{x}^{s'}$ is now presented, then the output is:

$$\begin{aligned}
(w \cdot \vec{x}^{s'})_i &= \sum_{j=1}^{N} \sum_{s=1}^{m} x_i^s x_j^s x_j^{s'} \\
&= \sum_{s=1}^{m} x_i^s (\vec{x}^s \cdot \vec{x}^{s'}) \\
&= \sum_{s=1}^{m} x_i^s \delta_{s,s'} \\
&= x_i^{s'},
\end{aligned} \tag{1.15}$$

and the stored state is perfectly recalled.

Let us now return to our analysis of the Hopfield model and examine the relaxation process when the states are not orthogonal. Given this, if we relax one of the stored states, we find in the first iteration, or in any iteration prior to which no neuron has changed its value:

$$\begin{aligned}
\mu_i^{s'} &\rightarrow \theta(\sum_{j=1}^{m} w_{ij} \mu_j^{s'}) \\
&= \theta(\sum_{s=1}^{m} \mu_i^s \sum_{j=1}^{N} \mu_j^s \mu_j^{s'}) \\
&= \theta(\sum_{s=1}^{m} \mu_i^s (\vec{\mu}^s \cdot \vec{\mu}^{s'})
\end{aligned}$$

$$= \theta(\, N\mu_i^{s'} + \sum_{s \neq s'} \mu_i^s(\vec{\mu}^s \cdot \vec{\mu}^{s'})\,). \tag{1.16}$$

Note that if the second term in the argument, $\sum_{s \neq s'} \mu_i^s(\vec{\mu}^s \cdot \vec{\mu}^{s'})$, is larger in magnitude than $N\mu_i^{s'}$ and of opposite sign, then the spin will be flipped and the stored state s' will be unstable. This second term, arising from state to state interaction, thus, can be viewed as "noise", which competes with the "signal", $N\mu_i^{s'}$. It is this "noise" which causes the standard Hopfield model to function so poorly. As a very crude estimate of this "noise" term, note that:

$$|\sum_{s=1}^{m} \mu_i^s(\vec{\mu}^s \cdot \vec{\mu}^{s'})\,| \leq (m-1)N. \tag{1.17}$$

As an example, with m = .1N and N = 100, this bound is 9N. Given this, it is easy to see how random fluctations in the selection of the stored states initially could easily generate a "noise" term large enough to swamp the "signal."

It is logical to ask how the above analysis effects error correction. We are led to consider, therefore, the relaxation process for a more general state, that is, for one of the stored states with some error:

$$\vec{\mu} = \vec{\mu}^{s'} + \vec{e}. \tag{1.18}$$

Here, $\vec{\mu}^{s'}$ is the target state and $\vec{e}$ is the error in the state at some point during the relaxation process. With this definition, if the ith neuron is the next neuron selected by the updating procedure, then, we have:

$$\mu_i \rightarrow \theta(\sum_{j=1}^{N} w_{ij}\mu_j)$$

$$= \theta(\sum_{j=1}^{N}\sum_{s=1}^{m} \mu_i^s\mu_j^s(\mu_j^{s'} + e_j)\,)$$

$$= \theta(\, N\mu_i^{s'} + \sum_{s \neq s'} \mu_i^s(\vec{\mu}^s \cdot \vec{\mu}^{s'}) + \sum_{s=1}^{m} \mu_i^s(\vec{\mu}^s \cdot \vec{e})\,). \tag{1.19}$$

The situation is more complicated than before because there are now two "noise" terms:

$$\eta = \eta_1 + \eta_2,$$

$$\eta_1 = \sum_{s \neq s'} \mu_i^s (\vec{\mu}^s \cdot \vec{\mu}^{s'}),$$

$$\eta_2 = \sum_{s=1}^{m} \mu_i^s (\vec{\mu}^s \cdot \vec{e}). \tag{1.20}$$

$\eta_1$ is just the state-state interaction due to the non-orthogonality of the states, as previously discussed. $\eta_2$ is the interaction of the error vector with all of the states. This more complicated "noise" term competes with the "signal" and is responsible for the poor error-correction ability of the standard Hopfield model.

In light of the above analysis, we now want to examine how Potter's modification algorithm helps to improve the standard Hopfield model and point out some of the reasons for its limitations. We consider the version of Potter's algorithm which tests the stored states. To begin with, we demonstrate the effect of a single modification and then discusss the consequences when multiple modifications are effected.

A single modification due to the relaxation of the stored state s' will yield:

$$w_{ij} \rightarrow w_{ij}^* = w_{ij} + \alpha(\mu_i^{s'} - \mu_i^{r,s'})\mu_j^{s'}(\mu_j^{s'} + 1) \tag{1.21}$$

Now, let us see the effect on the relaxation of $\vec{\mu}^{s'}$ with the matrix $w_{ij}^*$. We examine a step in the relaxation process before any neuron has changed its value and find that, now, if the ith neuron is sampled, the updating procedure will give:

$$\mu_i^{s'} \rightarrow \theta(\sum_{j=1}^{N} w_{ij}^* \mu_j^{s'})$$

$$= \theta(\ N\mu_i^{s'} + \eta(\{\vec{\mu}^s\}) + \sum_{j=1}^{N} \alpha(\mu_i^{s'} - \mu_i^{r,s'})\mu_j^{s'}(\mu_j^{s'} + 1)\mu_j^{s'}\ ). \tag{1.22}$$

If we use the fact that $\mu_j^2 = 1$, then this reduces to:

$$\mu_i^{s'} \rightarrow \theta(\ \ N\mu_i^{s'} + \eta(\{\vec{\mu}^s\}) + (N + \sum_{j=1}^{N} \mu_j^{s'})\alpha(\mu_i^{s'} - \mu_i^{r,s'})\ \ ). \tag{1.23}$$

If the states are stored with a random but balanced code, [4] [5] then the term

[4] A balanced code is one with an equal number of 1's and -1's, or 0's (if the "off" state is represented by a 0).

[5] Experimentally, a random, balanced code performs 3 to 4 times better than a random

$\sum_{j=1}^{N} = 0$. We assume this is true and observe the relationship between the "signal" term, $N\mu_i^{s'}$, the "noise" term, $\eta(\{\vec{\mu}^s\})$, and the modification term, $N\alpha(\mu_i^{s'} - \mu_i^{r's'})$. Note that if $\mu_i^{r's'} = \mu_i^{s'}$, then the modification term is identically zero, which is the desired result, since this corresponds to the situation in which the "signal" was stronger than the "noise". If $\mu_i^{r's'} > \mu_i^{s'}$, then $\mu_i^{r's'} = 1$ and $\mu_i^{s'} = -1 \Rightarrow \eta(\{\vec{\mu}^s\}) > 0 > N\mu_i^{s'}$, and further: $\mid N\mu_i^{s'} \mid < \mid \eta \mid$, but $N\alpha(\mu_i^{s'} - \mu_i^{r's'}) < 0$, so that provided $\alpha$ is chosen large enough, i.e. $\alpha > \frac{\mid N - \mid \eta \mid \mid}{N \mid \mu_i^{s'} - \mu_i^{r's'} \mid}$. If $\mu_i^{r's'} > \mu_i^{s'}$, the same result will obtain provided $\alpha$ is again chosen larger than the same limit. This follows from the fact that $N\alpha(\mu_i^{s'} - \mu i^{r's'})$ has to have the same sign as the "signal" term and the opposite sign to that of the "noise" term, so that provided $\alpha$ is sufficiently large, i.e. above the stated limit, the ith neuron will be stable. We can gaurantee that the whole state will be stable if we chose:

$$\alpha > \frac{\mid N - \mid \eta \mid \mid}{N \, min_i \mid \mu_i^{s'} - \mu_i^{r's'} \mid}. \tag{1.24}$$

The limitations of this approach lie in the fact that modifications which aid in the stability of one stored state may hinder the stability of other stored states. Clearly, the fact that out to $N^2$ stable stored states have already been achieved with this algorithm attests to the fact that despite this competition of modifications, there is an overall averaging effect which achieves a substantial increase in the capacity to store stable memory states.

We have outlined this argument in terms of a "signal", which is the only term present for orthogonal memory states, a "noise", which is due to the lack of orthogonality, and a modification term. What the modification term achieves, then, is an *effective orthogonaliztion* of the states with respect to the non-linear updating procedure, by eliminating the effect of the noise. In a linear system, as pointed out earlier, we could create up to N perfectly recallable stored states. Because the updating procedure in the Hopfield model is nonlinear, an effective orthogonalization with respect to that nonlinearity can achieve well above the dimensionality.

As we have discussed earlier, however, error corresction is non-existent at $\sim N^2$. When states with error are input, $\eta(\{\vec{\mu}^s\}) \rightarrow \eta(\{\vec{\mu}^s\}, \vec{e})$, and the error term is apparently then so large for a number of the neurons that the effect of the modifications is inconsequential.

---

but unbalanced code

The same analysis applies if the second version of Potter's algorithm is used–that of probing all of the states one Hamming unit away from the stored states, except that in the modification term, $\vec{\mu}^{s'}$ and $\vec{\mu}^{r\bullet'}$ should now be thought of as the probe states and their respective relaxed states. A radius of attraction of one Hamming unit could be gauranteed for m just below the dimensionality, but no better. Again, it is the combination of the size of the "noise" term and the competition between the various modifications which causes such limitations to arise.

To provide an alternative perspective on the Potter algorithm, we consider it from the point of view of the energy functional. The energy of a state $\vec{\mu}$ is given by the quadratic form in 1.3. Suppose we use the version of Potter's algorithm in which the modifications are based on the relaxation of the stored states. Denote the energy of $\vec{\mu}$ before the modifications by $\xi^0_{\vec{\mu}}$. Then, after the modification procedure, we have:

$$\xi_{\vec{\mu}} = \xi^0_{\vec{\mu}} - \frac{1}{2}\sum_{ij}\mu_i \sum_{s=1}^{m}(\mu_i^s - \mu_i^{r\bullet})\mu_j^s(\mu_j^s + 1)\mu_j$$

$$= \xi^0_{\vec{\mu}} - \frac{1}{2}\sum_{s=1}^{m}\Big( (\vec{\mu}\cdot\vec{\mu}^s) - (\vec{\mu}\cdot\vec{\mu}^{r\bullet})\Big)\Big(\sum_{j=1}^{N}\mu_j + (\vec{\mu}^s\cdot\vec{\mu})\Big), \qquad (1.25)$$

using the fact that $(\mu_j^s)^2 = 1$. We will assume a balanced code for the stored states. Let us consider the energy of one of the stored states, s', and then examine the region of the hypercube nearby the state s'. For $\vec{\mu} = \vec{\mu}^{s'}$, the energy equation 1.25 reduces to:

$$\xi_{\vec{\mu}} = \xi^0_{\vec{\mu}^{s'}} - \frac{1}{2}\sum_{s=1}^{m}\Big( (\vec{\mu}^{s'}\cdot\vec{\mu}^s) - (\vec{\mu}^{s'}\cdot\vec{\mu}^{r\bullet})\Big)\Big( \vec{u}^{s'}\cdot\vec{\mu}^s)\Big), \qquad (1.26)$$

(where we have assumed a balanced code). Consider the factor $(\vec{\mu}^{s'}\cdot\vec{\mu}^s)$. It will obviously be largest for s=s'. The prefactor of this term is $N-(\vec{\mu}^{s'}\cdot\vec{\mu}^{r\bullet'})$ which is $\geq 0$ and is zero *iff* the state s' was stable before the modifications. Thus, if s' was unstable before the modification, this modification term will lower the energy of the state s'. This is depicted graphically here: Terms in the sum due to nearby stored states may in fact reverse this process, particularly if their density is high (perhaps $m \geq N^2$). This is possible because $\vec{\mu}^{r\bullet}$ for these nearby stored states will have a probability of being closer to $\vec{\mu}^{s'}$ than $\vec{\mu}^s$ is; when this situation arises, the prefactor

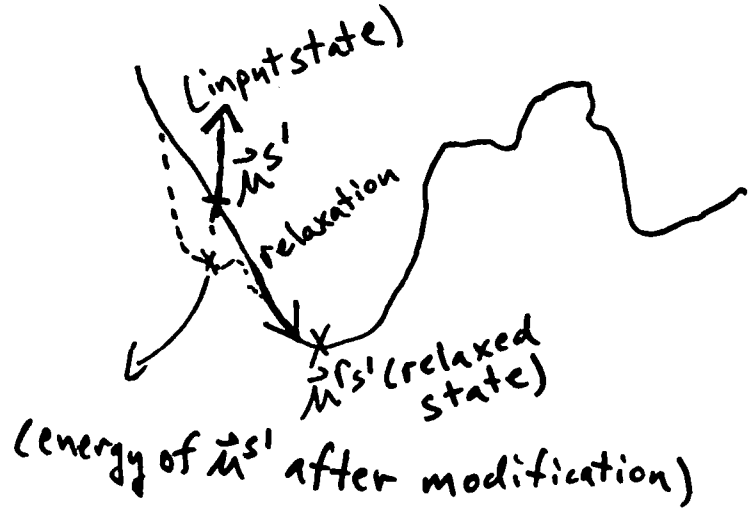



Figure 1.3: . Instability of state s' causes the largest modification term to contribute to lowering the energy of state s': dashed line

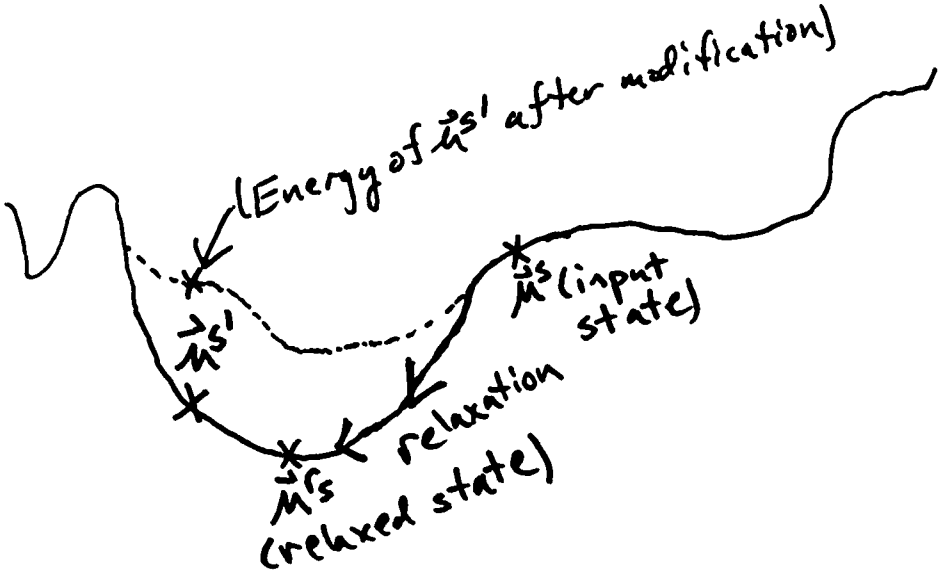



Figure 1.4: . The presence of nearby states can generate modification terms which tend to raise the energy of state s' as described in the text: dashed line

$(\vec{\mu}^{s'} \cdot \vec{\mu}^{s}) - (\vec{\mu}^{s'} \cdot \vec{\mu}^{r \cdot s})$ will be negative. This is depicted graphically for $\vec{\mu}^{s'}$ and one of its neighboring states $\vec{\mu}^{s}$: It is these competing nearby states which provide some upper bound $\geq N^2$ on the number of stable stored states obtainable, because if there are too many nearby neighbors, the algorithm can't lower the energy of the stored state. We caution, however, that this energy perspective has some limitations: the presence of a gap in the energy is only a neccesary and not a sufficient condition for the system to descend through that gap: the reason for this lies in the discrete nature of the jumps; the system can get caught.[6] [7]

The energy perspective if $\vec{\mu}$ is not one of the stored states but rather nearby one of the stored states is more complicated. This stems from both the competition of modification terms due to nearby states and from also from the fact that the largest term in the sum may now have the wrong sign: $\vec{\mu}$ may now be closer to $\vec{\mu}^{r \cdot s'}$ than $\vec{\mu}^{s'}$. In addition, modification terms due to other states targeting the same stored state may have the wrong sign for the same reason. As a consequence, we would expect the ability of Potter's algorithm to improve error-correction in the Hopfield model to be quite limited, and, indeed, as pointed out earlier, this is exactly what he has observed experimentally.

## 1.4 Some Alternative Energy Functions

Among possible methods of improving the Hopfield model, the embedding of different energy(Liapunov) functions via the updating procedure poses a promising alternative to iterative modifications procedures, like "unlearning" or Potter's algorithm, which aim at simply optimizing the matrix in the quadratic form of equation1.3. Among the options that we have considered is the possibility of adding a cubic term to the energy function, so that $\xi_{\vec{\mu}}$ would take the form:

$$\xi_{\vec{\mu}} = -\frac{1}{2}\sum_{i,j} w_{ij}\mu_i\mu_j - \frac{1}{3!}\sum_{i,j,k} \mu_i\mu_j\mu_k q_{ijk} \qquad (1.27)$$

[6] For this reason, one should not be misled by the fact that in the above schematic diagrams, we have drawn continuous lines connecting the states.

[7] The claim can be demonstrated by working through some simple three-dimensional examples of the Hopfield model. See the three-dimensional examples in the Appendix section of this report.

$$with\ w_{ij} = \sum_{s=1}^{m} \mu_i^s \mu_j^s \ \ and \ \ q_{ijk} = \sum_{s=1}^{m} \mu_i^s \mu_j^s \mu_k^s.$$

By choosing the associated updating procedure to be of the form:

$$\mu_i \rightarrow ( \sum_{j=1}^{N} w_{ij}\mu_j + \frac{1}{2} \sum_{jk} q_{ijk}\mu_j\mu_k ), \tag{1.28}$$

the algorithm is gauranteed to descend monotonically. The definition of $q_{ijk}$ is chosen to be analogous to that of $w_{ij}$: it corresponds to a measure of the correlation between all triples of neuronal values $\mu_i$, $\mu_j$, and $\mu_k$ in the stored states $s : 1 \cdots m$. While this modified energy function, may prove to work better than the original model, the amount of additional computational complexity involved in the relaxation process may not be worth the gain. Further, modifying the quadratic and cubic tensors also requires a quite significantly longer amount of time if the updating procedure is involved in the optimization. Finally, there is some question as to whether the form of the cubic order coupling constant is really the best form. The question is raised because of the fact that if the ith, jth, and kth neurons all have the value -1 in a given state then their product is, of course, negative: this contradicts the notion of correlation as Potter has pointed out.[8]

Another alternative, which has proved to be very promising in some preliminary investigations in three dimensions, [9] is the following energy functional:

$$\xi_{\vec{\mu}} = -\frac{1}{2} \sum_{s=1}^{m} (\vec{\mu} \cdot \vec{\mu}^s)( (\vec{\mu} \cdot \vec{\mu}^s) + \alpha \sum_{s \neq s'} (\vec{\mu}^s \cdot \vec{\mu}^{s'}) ) \tag{1.29}$$

The first term is the original Hopfield term; the second term contributes most when nearby states are close. This can be rewritten as:

$$\xi_{\vec{\mu}} = -\frac{1}{2} \sum_{ij} ( \mu_i w_{ij} \mu_j + \alpha \mu_i Q_{ij} ), \tag{1.30}$$

where:

$$Q_{ij} = \sum_{s=1}^{m} \Big( \sum_{s=1}^{m} \mu_i^s \sum_{s' \neq s} \mu_j^s \mu_j^{s'} \Big). \tag{1.31}$$

[8] T. Potter, personal communication

[9] See the three-dimensional examples in the Appendix

A descent algorithm can be gauranteed if we choose:

$$\mu_i \rightarrow \theta(\sum_{j=1}^{N} w_{ij}\mu_j + \frac{\alpha}{2}Q_{ij}). \tag{1.32}$$

This ammounts to a special recipe for choosing biases, since:

$$\phi_i = \frac{\alpha}{2}\sum_{j=1}^{N} Q_{ij} \tag{1.33}$$

is just an offset. Hopfield[1] employed a special choice of biases in his original model with spin values 0 and 1 and found that it made the model perform as well as when the model was run with spins $\pm 1$, and no offset; no formula for the choice of biases was given in his article, however. We are suggesting that a special choice of biases for spins (+/-)1 could improve matters even more,[10] and are proposing the above form as an appropriate means of making this choice. In the three-dimensional examples which we have worked in the Appendix, we found that $\alpha$ must be chosen as a function of the stored states: [11] [12]

$$\alpha = f(\{\vec{\mu}^s \cdot \vec{\mu}^{s'}\}). \tag{1.34}$$

[10] See Appendix.

[11] For the particular examples worked in three dimensions with two stored states, $\vec{\mu}^1$ and $\vec{\mu}^2$, we found that $\alpha = 5(\vec{\mu}^1 \cdot \vec{\mu}^2) - 1$.

[12] The explicit form has yet to be derived, but our explorations in three dimensions in the Appendix suggest that a good choice should probably be a function of how close the nearest neighbors are. For this reason we have chosen the form above.

# Chapter 2

# The Analog Hopfield Model, and Future Directions with Continuum Models

## 2.1 The Analog Hopfield Model

In subsequent articles following the appearance of Hopfield's 1982 article, Hopfield has developed the idea of a continuous version of his original model.[3,4] [5][ , ] In this analog version, Hopfield defines $u_i$ to be the input to the ith neuron and $V_i = g(u_i)$ to be the output ($g(u_i)$ is a sigmoid) of the ith neuron. The analog version is composed of an R-C network constructed from standard op-amp's, [3,4] the dynamics of which are governed by:

$$C_i(\frac{du_i}{dt}) + \frac{u_i}{R_i} = \sum_{j=1}^{N} w_{ij}V_j + I_i \tag{2.1}$$

(here, $I_i$ is an external current, which is just an offset). As in the digital version of the model, the system descends monotonically to a minimum of

an associated Liapunov function: [1]

$$\xi = -\frac{1}{2}\sum_{i,j} w_{ij}\mu_i\mu_j + \sum_{i=1}^{N}\left(\frac{1}{R_i}\int_0^{V_i} g_i^{-1}(V)dV - \sum_{i=1}^{N} I_i V_i\right). \tag{2.2}$$

Comparing the performance of the analog version of the model with that of the digital version, Potter [7] has found that the analog model functions better than the digital model in its ability to store stable stored states and do error correction. Likewise, Hopfield [5] has found that his analog model arrives at vastly better solutions of the *travelling salesman problem.* Still, there is room for much improvement.

[1] For a proof, see [3]

## 2.2 Alternative Analog Equations

Given the limitations of the Hopfield model, both analog and digital, we have begun exploring the problem of stable stored states and error-correction in a more general way at the suggestion of Professor Cooper. As a starting place, we began by studying equations of the form:[2]

$$\frac{du}{dt} = -\nabla_u F(u;\{u^s\}) \tag{2.3}$$

$$with \;\; F(u;\{u^s\}) = \prod_{s=1}^{m}(u-u^s)^2. \tag{2.4}$$

In the simplest situation, in one dimension, with one stored state, we have:

$$\frac{du}{dt} = -2(u-u^1). \tag{2.5}$$

The solution is easily found:

$$u = u^1 + (u(0)-u^1)e^{-2t} \tag{2.6}$$

$$with \;\; lim_{t\to\infty}u(t) = u^1. \tag{2.7}$$

There is only one fixed point, the stored state, and it is stable. Furthermore, every state not beginning at $u^1$ converges to it asymptotically.

. For two stored states in one dimension, the equation of motion is:

$$\frac{du}{dt} = -2(u-u^1)(u-u^2)(u-\frac{u^1+u^2}{2}). \tag{2.8}$$

The fixed points of the system are $(u=u^1)$, $(u=u^2)$, and $(u=\frac{u^1+u^2}{2})$. Again as desired, only the stored states themselves are stable fixed points, and they are the only attractors, partitioning the space evenly. For one-dimension, with three or more stored states, the original form of F leads to an equation of motion with a non-uniform partitioning of the region of attraction of the stored states. For example, with three stored states, we have:

$$\frac{du}{dt} = -2(u-u^1)(u-u^2)(u-u^3)\cdot\sum_{i\neq j}\frac{1}{2}(u-u^i)(u-u^j). \tag{2.9}$$

[2]This form was suggested by Amir Dembo and Ofer Zeitouni, Visiting Assistant Professors(Res.), in Applied Mathematics, Brown University.

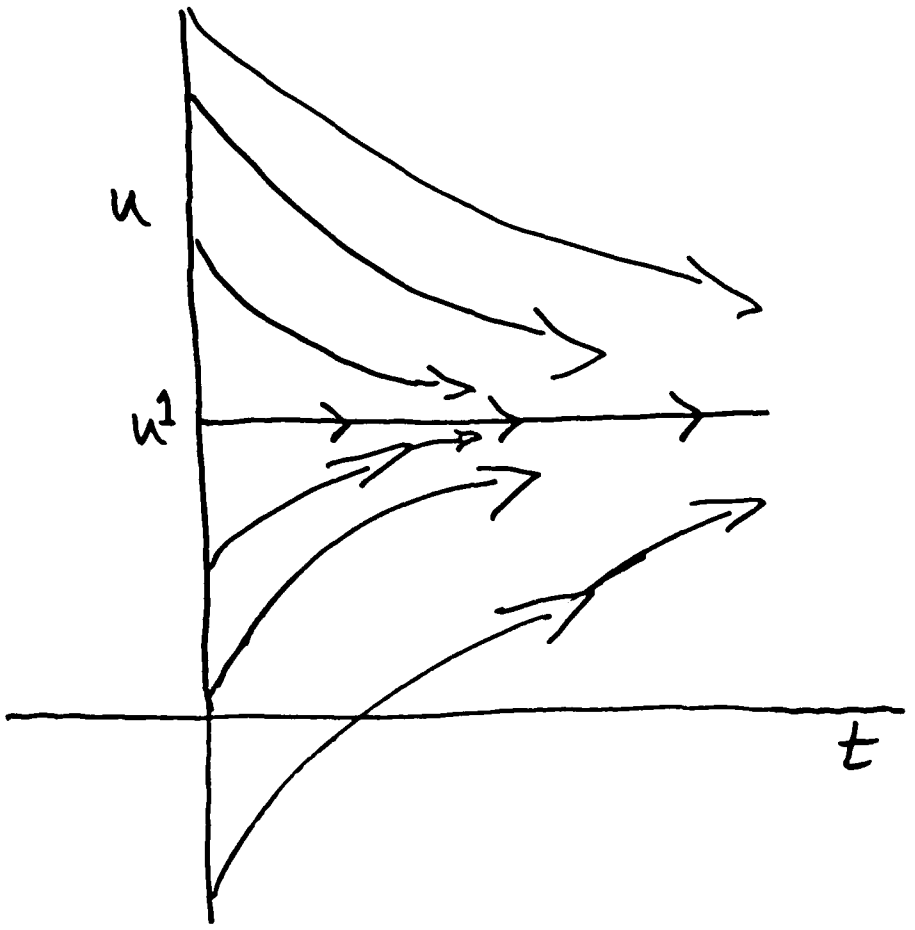



Figure 2.1: Phase diagram for one stored state in one dimension.

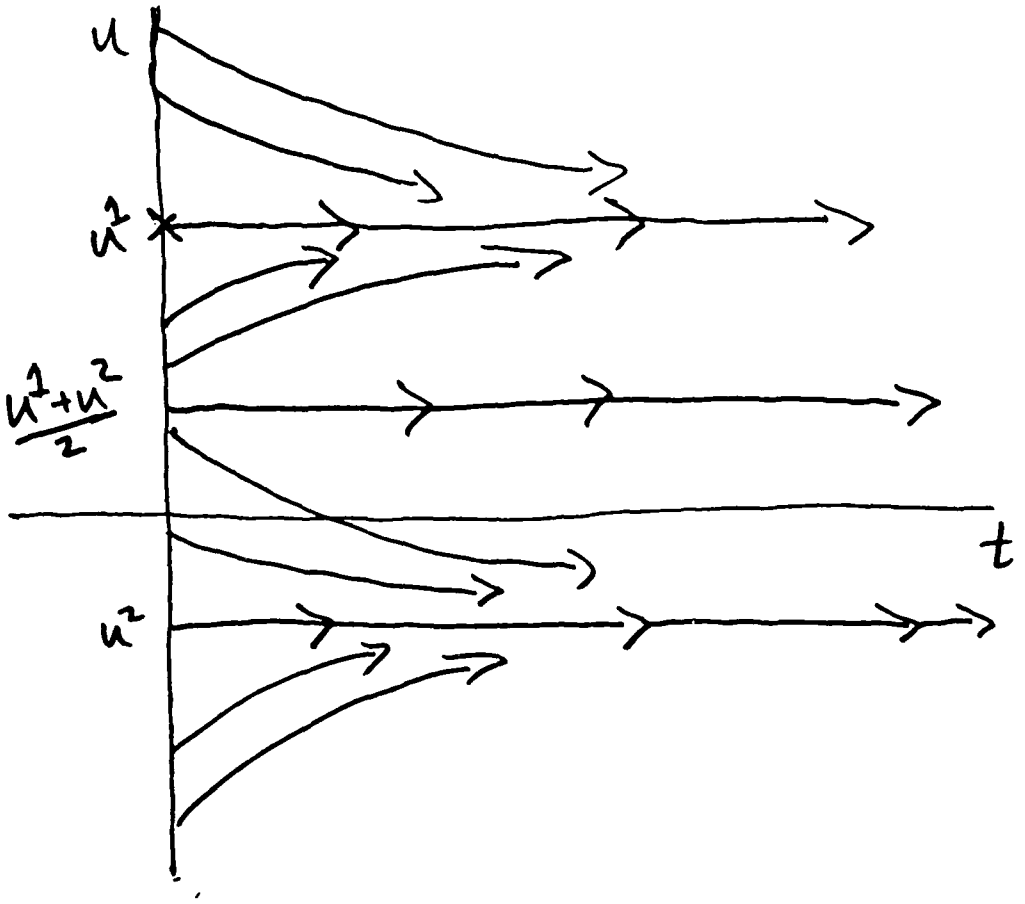



Figure 2.2: Phase diagram for two stored states in one dimension.

The unequal partitioning comes from the last factor involving the $\sum_{i \neq j}$, which is an interference term due to the "interaction" of the stored states.

This observation led us to modify the original form of F, or more importantly $\nabla F$, so that the domain of attraction would be evenly partitioned among the stored states, while at the same time, so that the stored states would remain the only stable fixed points and attractors in the system. The beauty of the result we have obtained below is that its properties are independent of the number, density, or distribution of the stored states. The form of $\nabla F$ satisfying the above requirements is:

$$\nabla F = \alpha(u - u^1) \cdots (u - u^m)(u - \frac{u^1 + u^2}{2}) \cdots (u - \frac{u^{m-1} + u^m}{2}), \qquad (2.10)$$

so that the equation of motion becomes:

$$\frac{du}{dt} = -\nabla F = -\alpha(u - u^1) \cdots (u - u^m)(u - \frac{u^1 + u^2}{2}) \cdots (u - \frac{u^{m-1} + u^m}{2}). \qquad (2.11)$$

The phase diagram for this is:

If we now consider the problem in higher dimensions, we observe that if we take the coordinates to be independent of one another we can generate

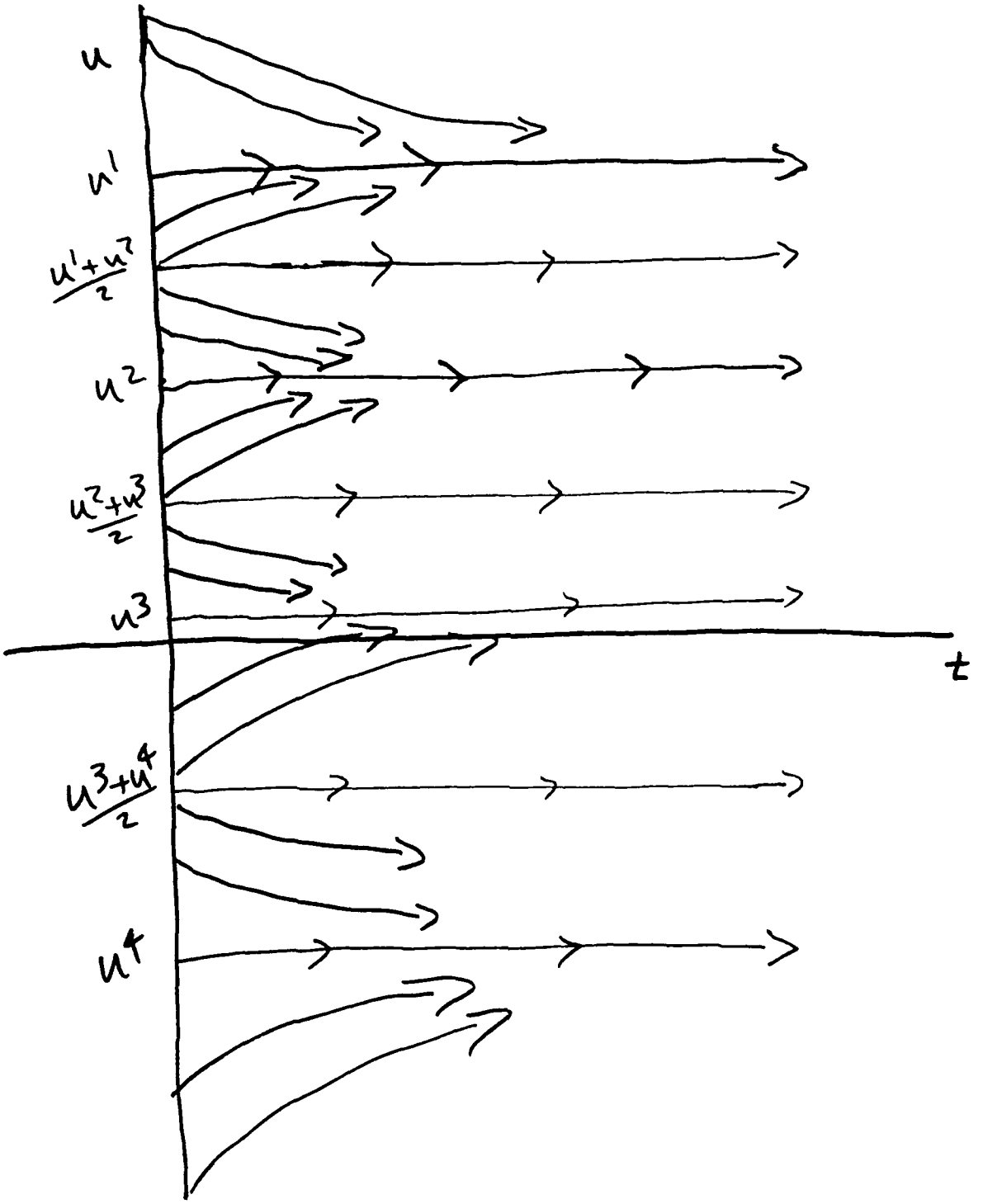
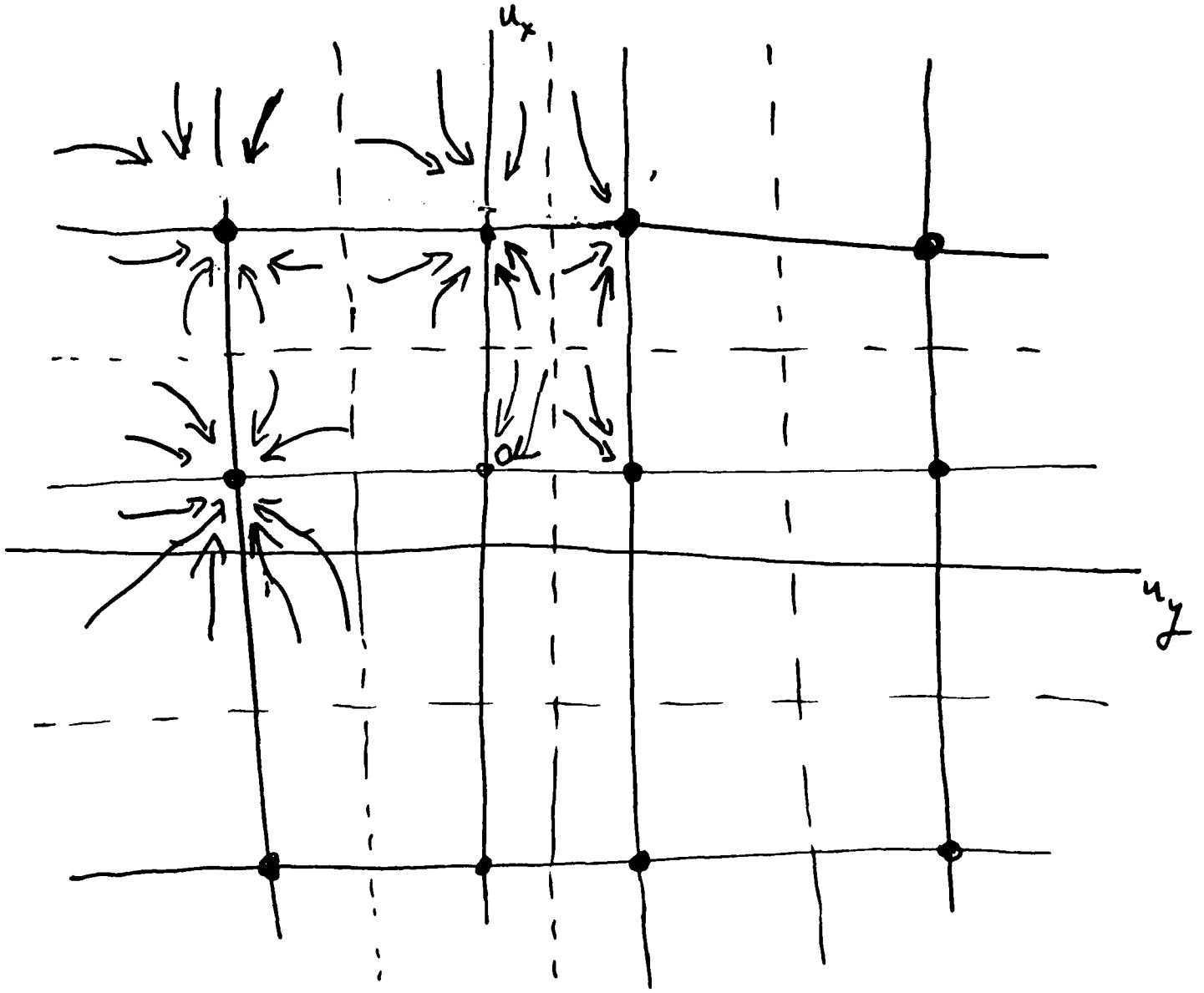



Figure 2.3: Phase diagram in one dimension for a modified $\nabla F$; all of the sytems properties are maintained regardless of the number, density, or distribution of the stored states.

a hyper-grid of stored states which are the only stable fixed points and attractors, and which partition the hypersphere into hyperrectangles of attraction, by simply adding the coordinate index as a subscript in equation 2.11.



# Appendix

Three Dimensional Examples Comparing the Digital Hopfield Model with Various Algorithms to Improve It

In this appendix, we discuss three examples of the Hopfield model in three dimensions. In the first three figures, we show how the standard Hopfield model functions in each of the three examples. Examples 2 and 3 are then revisited with after Potter's modification has been applied to the synaptic matrix. Finally, examples 2 and 3 are studied after the standard Hopfield model has been modified by choosing biases according to the recipe that we specified in chapter 2, and the results are contrasted with those of Potter's algorithm and the standard model.

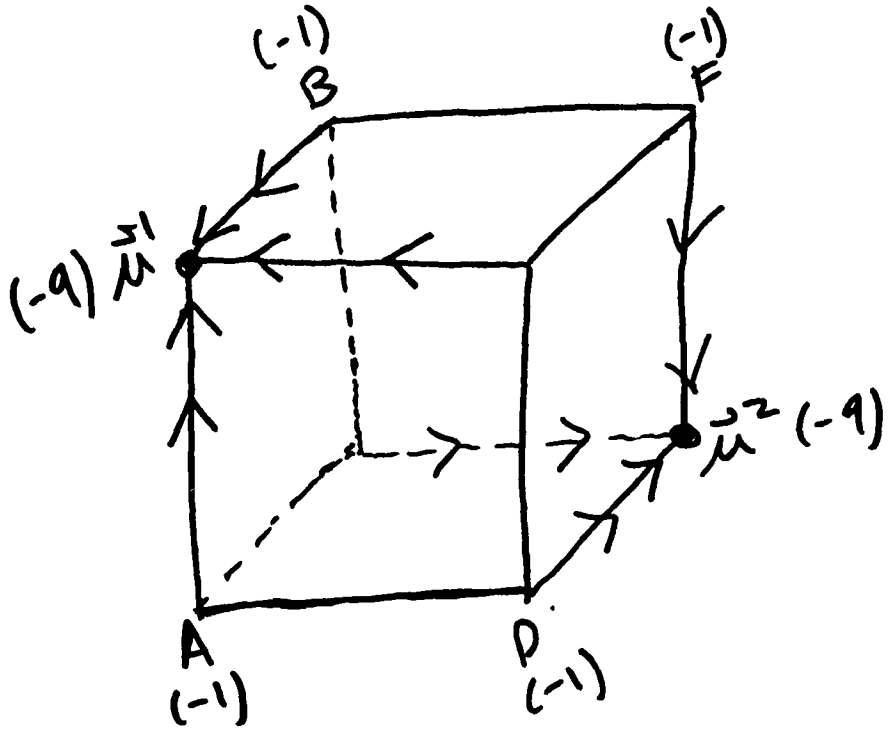



Figure .1( fig a.1)The standard Hopfield model when the states are well separated. Energies are in parenthesis;arrows indicate flows. The situation is ideal.

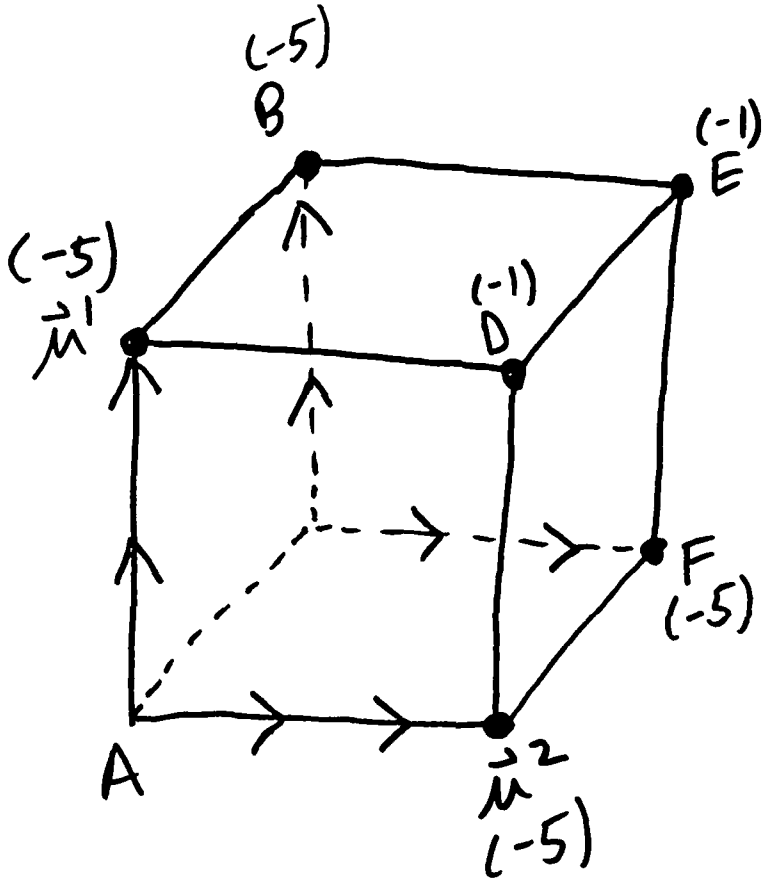



Figure .2 ( fig a.2 ) States closer together in the Standard Hopfield. Spurious attractors and fixed points appear; flows are no longer ideal. There is "noise" due to state-state interaction and state-error interaction. All dots are fixed points.

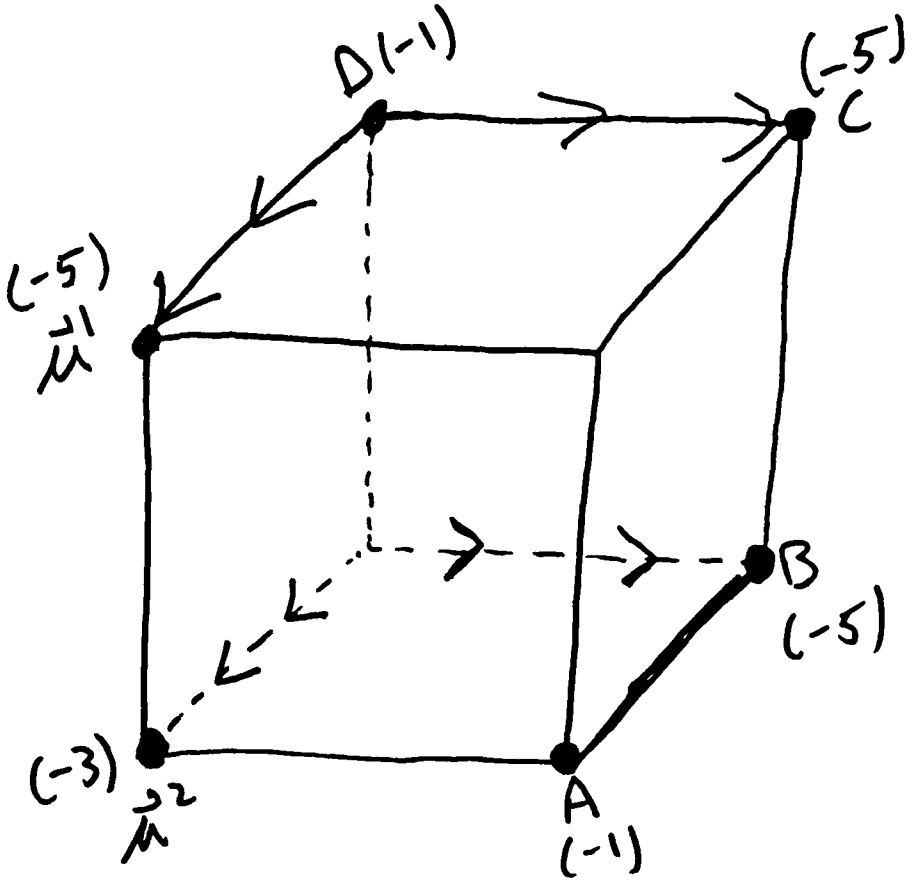



Figure .3( fig a.3)Standard Hopfield$_4$ model. The states are placed even closer together. Spurious attractors and fixed points. Note also, as in the previous figure, merely the presence of an energy gap is not enough to cause a flow (see point F in this figure and points D and E in the previous figure).

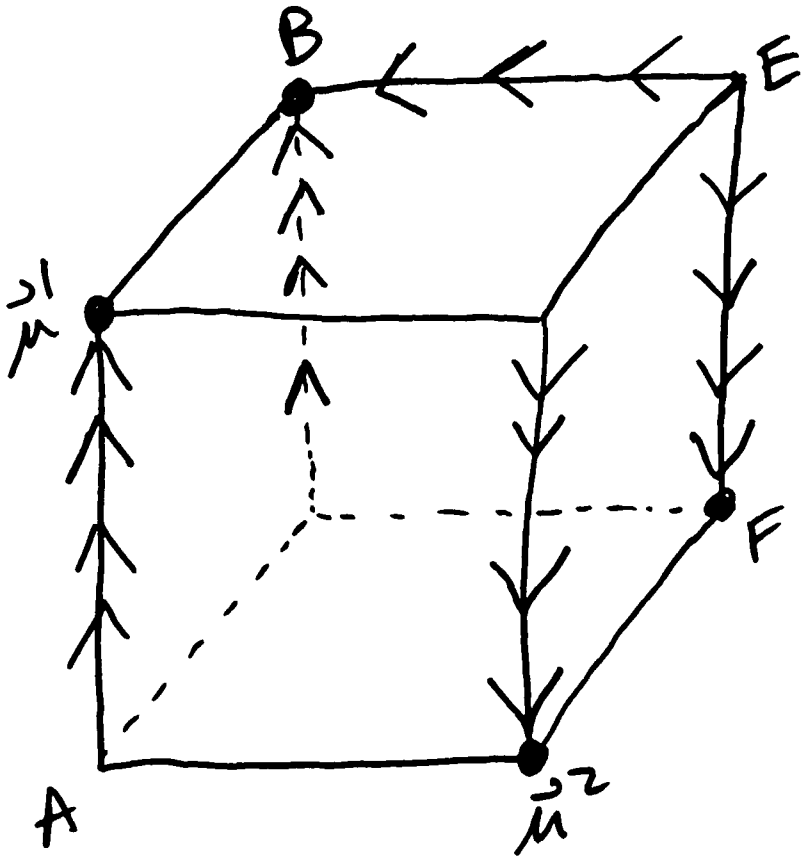



Figure .4( fig a.4)Ex. 2 revisited with Potter's algorithm. Improvements: 4 flows of nearest neighbors, instead of just 2, are correct. Spurious attractors and fixed points still persist.

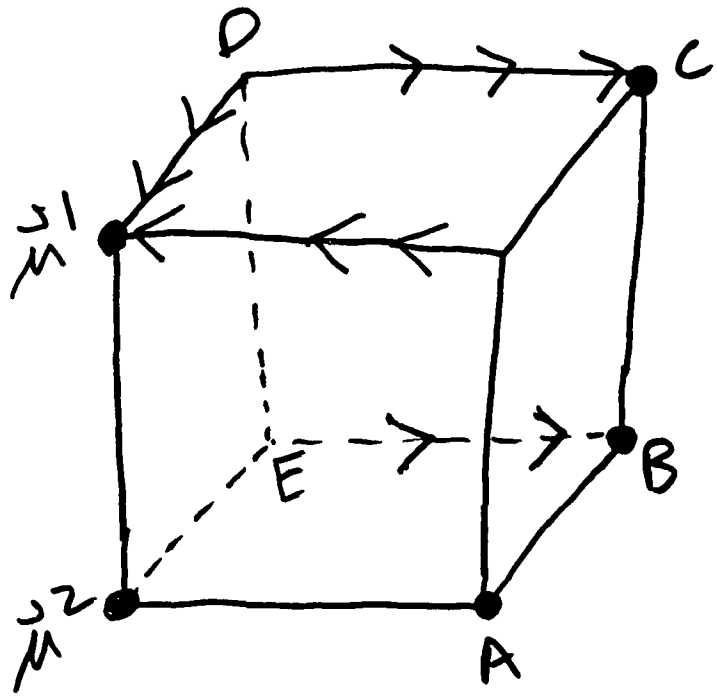



Figure .5 ( fig a.5 ) Ex.3 revisited with Potter's algorithm. No improvement over the standard model.

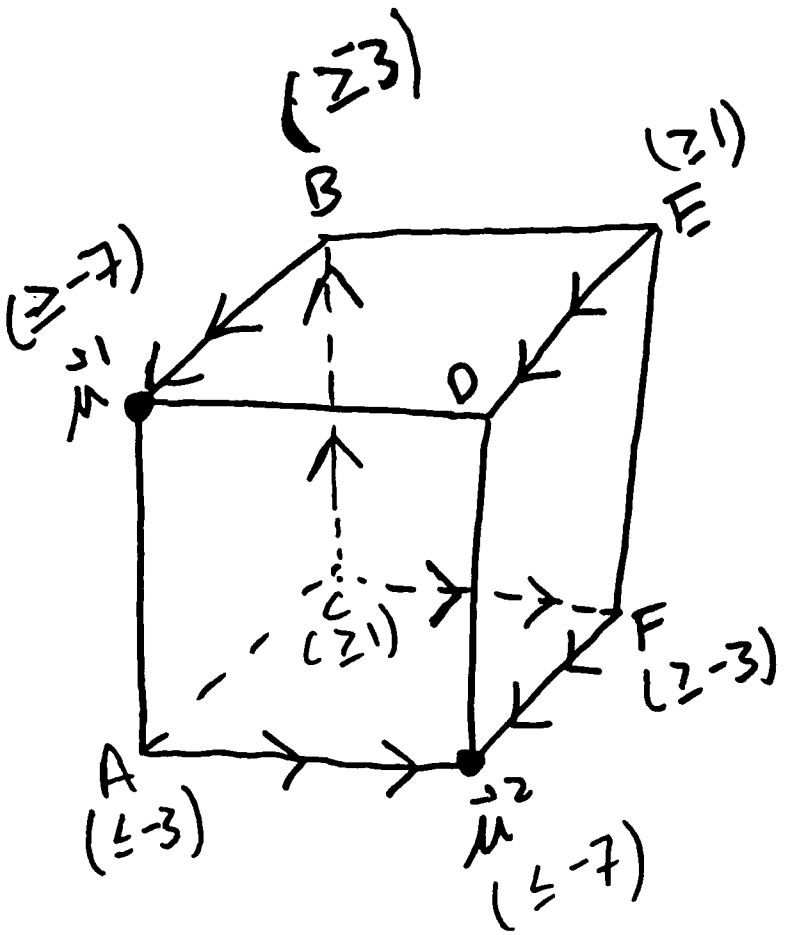
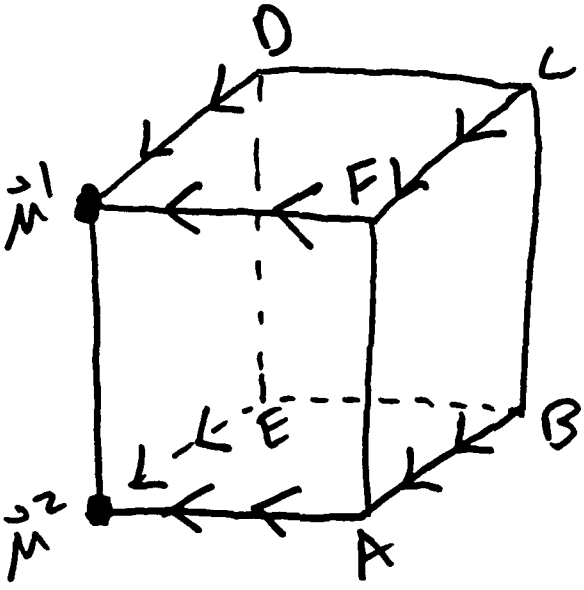
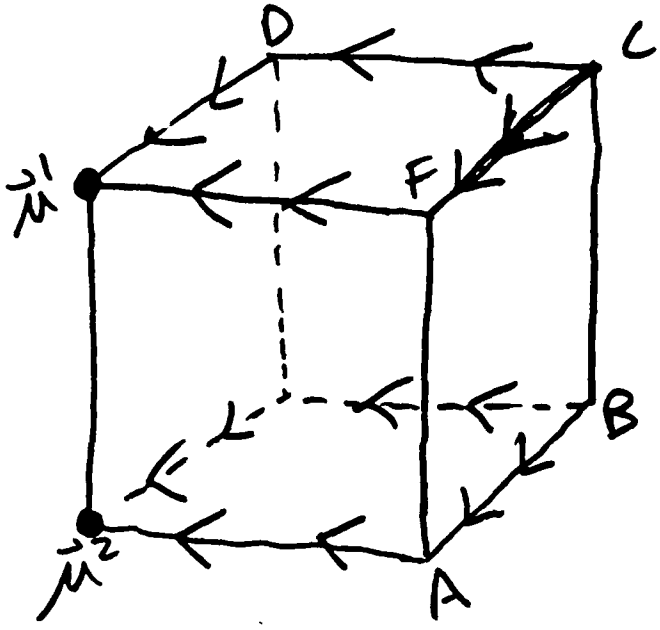



Figure .6( fig a.6) Ex.2 revisited with special choice of biases as defined in chapter 2. Great improvement: all but one nearest neighbor exhibits the right flow, and all but one next nearest neighbor flows correctly. $\alpha \leq -2$.




($\alpha = 4$)




($\alpha > 4$)

Fig 7(a.7) Ex. 3 Revisited with special choice of biases as defined in Chap. 2. The situation is ideal. Nearest and next nearest neighbors flow correctly. $\alpha \geq 4$.

# Bibliography


[1] J.J. Hopfield, *Proc. Natl. Acad. Sci. USA, Vol 79, pp.2554-2558, April 1982, Biophysics*

[2] J.J. Hopfield, *"Unlearning has a Stabilizing Effect in Collective Memories", Nature, Vol. 304, July 14, 1983*

[3] J.J. Hopfield, *Proc. Natl. Acad. Sci. USA, Vol.81,pp.3088-3092,May 1984, Biophysics*

[4] J.J Hopfield and D.W. Tank, *Biol. Cybern. 52, 141-152, 1985*

[5] J.J. Hopfield and D.W. Tank, *Science. Vol. 233, Aug. 8, 1986, pp.625-633*

[6] T. Kohonen,*IEEE Trans. on Comp., April 1974, pp. 444-5.*

[7] T. Potter, Ph.d. Thesis,S.U.N.Y Binghampton,Dept. of Comp. Sc., in preparation

END

3-87

DTIC